IN THE UNITED STATES PATENT AND TRADEMARK OFFICE

Applicant:	Avadhanam et al.	Examiner:	Pham, Khanh B.
Serial No.:	10/830,164	Group Art Unit:	2166
		Confirmation No.:	8149
Filed:	April 21, 2004	Docket No.:	MS167378.02/40062.128USC1
Title:	METHOD AND SYSTEM FOR CREATING A DATABASE TABLE INDEX USING MULTIPLE PROCESSING UNITS		

CERTIFICATE UNDER 37 CFR 1.8:

I hereby certify that this correspondence is being transmitted via EFS-Web to the U.S. Patent Office on January 15, 2008.

By: Krystle Henley
Name: Krystle Henley

SUPPLEMENTAL AMENDMENT

Mail Stop Amendment
Commissioner for Patents
P.O. Box 1450
Alexandria, Virginia 22313-1450

Dear Sir:

In response to the Office Action mailed June 28, 2007, please amend the above-identified application as follows:

Amendments to the Claims are reflected in the listing of claims that begins on page 2 of this paper.

Remarks begin on page 9 of this paper.

Amendments to the Claims:

This listing of claims will replace all prior versions and listings of claims in the application.

Listing of Claims:

Listing of Claims:

1. (Currently Amended) A method of creating an index for a database table of records, the method occurring in a computer environment having a plurality of processing units wherein each processing unit has access to the table, the method comprising the steps of:

determining partition delimiters, each partition delimiter separating the table into non-overlapping partitions of records, each partition dedicated to one processing unit for index creation, wherein the step of determining comprises sampling the database table of records to determine an approximate distribution of at least one key value in the records;

accessing the table records in parallel, wherein each processing unit accesses each of the records, wherein the step of accessing occurs after the step of determining;

filtering the accessed records in parallel, wherein each processing unit determines which records to keep;

independently creating a plurality of sub-indexes, wherein at least two sub-indexes are created by different processing units;

merging the sub-indexes together to create a final index related to the table; and

storing the final index for later use in locating records.

2. (Original) A method as defined in claim 1 wherein the act of creating the sub-indexes further comprises sorting the records and generating a data structure based on the sorted records.

3. (Original) A method as defined in claim 2 wherein the data structure is a B-Tree data structure.

4. (Original) A method a defined in claim 2 wherein the data structure has multiple levels.

5. (Original) A method as defined in claim 2 wherein the data structure is a clustered index.

6. (Original) A method as defined in claim 1 further comprising gathering sub-index statistical information and stitching sub-index statistical information.

7. (Original) A method as defined in claim 1 wherein the method is initiated by and index creation manager module.

8. (Original) A method as defined in claim 1 wherein the method is initiated by a query manager in response to a supplied query.

9. (Original) A method as defined in claim 1 wherein the method initiated automatically in response to a modification to the table.

10. (Currently Amended) A method as defined in claim 1 wherein the act of determining partition delimiters comprises:

~~sampling the table records to determine an approximate distribution of the values in a key field;~~

creating a histogram based on the sampled information; and

evaluation the histogram to determine the partition delimiters.

11. (Original) A method as defined in claim 10 further comprising:

determining a processing unit goal value based on the number of processing units in the computer system;

determining a least common multiple value based on the processing unit goal value;

determining whether the histogram information may be substantially evenly split into the least common multiple value number of partitions;

if so, creating the partition delimiters based on the least common multiple value; and

if not, adjusting the processing unit goal to determine a new least common multiple value to determine partition delimiters.

12. (Previously Presented) A computer storage medium readable by a computer and encoding instructions for executing the method recited in claim 1.

13. (Previously Presented) A computer storage medium readable by a computer and encoding instructions for executing the method recited in claim 11.

14. (Currently Amended) A system for database table index creation for a database table, the database table stored in memory and comprising a plurality of records, the system comprising:

a partition tool that determines partition delimiters, each partition delimiter separating the table into non-overlapping partitions of records, each partition dedicated to one processing unit for index creation, wherein the step of determining comprises sampling the database table of records to determine an approximate distribution of at least one key value in the records;

a plurality of processing units that respectively accesses the database table in parallel, wherein each of the respective processing units accesses each of the records and filters the accessed records to determine which records to keep and wherein each of the respective processing units creates a sub-index of database table records resulting in a plurality of sub-indexes;

a merge tool that merges the plurality of sub-indexes into a final database table index; and

a store tool that stores the final database table index for later use in locating records.

15. (Original) A system as defined in claim 14 wherein each processing unit further comprises:

a scanning module that scans the database table;

a filter module that filters the accessed records and selectively predetermined records; and

a sorting module that sorts records kept by the filter module into a sub-index.

16. (Original) A system as defined in claim 15 wherein the scanning module, filter module and sorting module, for each processing unit, operates concurrently.

17. (Original) A system as defined in claim 15 further comprising a sampling module for sampling the database table and a partition module for dividing the records into substantially equal quantities related to the number of processing units.

18. (Currently Amended) A method of creating an index for a database table of records, the method occurring in a computer environment having a plurality of processing units wherein more than one processing unit has access to the table, the method comprising the steps of:

determining partition delimiters, each partition delimiter separating the table into non-overlapping partitions of records, wherein the step of determining comprises sampling the database table of records to determine an approximate distribution of at least one key value in the records, and wherein at least one partition is dedicated to a first processing unit for index creation and at least one other partition is dedicated a second processing unit for index creation;

the first processing unit accessing a table record and determining whether the table record is associated with the at least one partition dedicated to the first processing unit;

the first processing unit only processing the accessed table record when the accessed table record is associated with the at least one partition dedicated to the first processing unit; and

storing a result produced by the first processing unit for later use in locating records.

19. (Original) A method as defined in claim 18 further comprises:

upon determining that the accessed table record is not associated with the at least one partition dedicated to the first processing unit, passing the accessed record to the second processing unit for index creation.

20. (Currently Amended) A method of creating an index for a database table of records, the method occurring in a computer environment having a plurality of processing units wherein more than one processing unit has access to the table, the method comprising:

determining partition delimiters, each partition delimiter separating the table into non-overlapping partitions of records, each partition dedicated to one processing unit for index creation, wherein the step of determining comprises sampling the database table of records to determine an approximate distribution of at least one key value in the records;

accessing the table records in parallel, wherein each processing unit accesses each of the records, wherein the step of accessing occurs after the step of determining;

filtering the accessed records in parallel, wherein each processing unit determines which records to keep;

independently creating a plurality of sub-indexes, wherein at least two sub-indexes are created by different processing units;

allocating blocks of a disk to store each sub-index, wherein parts of each sub-index are stored on consecutive blocks on the disk;

merging the sub-indexes together to create a final index related to the table; and

storing the final index for later use in locating records.

21. (Original) A method as defined in claim 20 wherein the act of allocating portions of the disk allocates a predetermined number of blocks, the predetermined number of blocks is determined during the determination of the partition delimiters.

22. (Original) A method as defined in claim 20 wherein the allocation of portions of the disk comprises:

maintaining a cache of allocated pages and allocating pages for each partition in the cache for each processing unit; and

retrieving a pre-determined number of database pages upon request, and wherein the number of pages to allocate upon each request is determined by the size of the cache.

23. (Original) A method as defined in claim 22 wherein the cache has a size depending on the size of the index being built and the number of currently available free pages in the system.

24. (Currently Amended) In a computer system having a plurality of processing units, an index creation system for creating an index of information for a table of data records, the system comprising:

a sampling module that samples the table of data records to determine sub-index delimiters, wherein the sub-index delimiters are used as partition delimiters separating the table into non-overlapping portions of records;

two or more index creation modules, each index creation module associated with a processing unit, each index creation module creates a sub-index resulting in a plurality of sub-indexes;

a merge module that merges the sub-indexes into a final index,

wherein each index creation module comprises:

an access module that accesses each of the data records from the table of data records;

a filter module that filters data records according the sub-index delimiters to keep only relevant data records; and

a sorting module that sorts the relevant data records into a sub-index; and

a store module that stores the final index for later use in locating records.

25. (Original) A system as defined claim 24 further comprising a memory allocation module that allocates parts of memory for storing the sub-indexes, and wherein the memory allocation module allocates a predetermined number of parts, the predetermined number of parts is determined during the determination of the delimiters.

26. (Original) A system as defined in claim 24 further comprising a cache memory module that manages a cache of allocation pages and allocates pages for storing each sub-index in the cache and wherein the number of pages allocated to the cached is determined upon determining the delimiters.

27-28. (Cancelled)

REMARKS

This Supplemental Amendment and the following remarks respond to the Office Action mailed June 28, 2007. In addition, this Amendment and the following remarks are made in response to the in person interview conducted on December 14, 2007. The remarks made in this amendment are intended to replace the remarks made in Applicants' previous response filed on November 28, 2007. Applicants withdraw all remarks made in the previous Amendment filed on November 28, 2007 that are not reproduced or incorporated by reference in the current Amendment. Claims 1-28 stand rejected under 35 U.S.C. § 103(a) as being unpatentable over Gupta et al., US Patent No. 6,438,562, hereinafter "Gupta," in view of Blank et al., US Patent No. 5,842,208, hereinafter "Blank."

Reconsideration of these rejections, as they might apply to the original and amended claims in view of these remarks, is respectfully requested.

In this Response, claims 1, 10, 14, 18, 20, and 24 have been amended, claims 27 and 28 have been cancelled, and no claims have been added. Therefore, claims 1-26 remain present for examination.

Summary of Interview

Applicants would like to thank Examiner Pham for his comments during the in person interview conducted on December 14, 2007. During the interview, Examiner Pham and the Applicants' representative discussed the Gupta and Blank references and possible claim amendments. Examiner Pham agreed that the previous amendments overcome the § 101 rejection for the reasons stated in Applicants response filed on November 28, 2007. No agreement was reached with respect to the §103 rejection of claims 1-28 over Gupta in view of Blank.

Claim Rejections – 35 U.S.C. § 101

As discussed, the Examiner agreed that Applicants' Amendment of November 28, 2007 overcame the rejections of the pending claims under § 101.

Claim Rejections – 35 U.S.C. § 103(a)

Claims 1-26 stand rejected under 35 U.S.C. § 103(a) as being unpatentable over Gupta in view of Blank. Applicants respectfully traverse the § 103(a) rejections because either the Examiner failed to state a *prima facie* case of obviousness or the current amendments to the claims now render the Examiner's arguments moot. To establish a *prima facie* case of obviousness under 35 U.S.C. § 103(a), the references must teach or suggest all of the claimed limitations to one of ordinary skill in the art at the time the invention was made. M.P.E.P §§ 2142, 2143.03; *In re Royka*, 490 F.2d 981, 985 (C.C.P.A. 1974); *In re Wilson*, 424 F.2d 1382, 1385 (C.C.P.A. 1970). Further, under *KSR Int'l Co. v. Teleflex, Inc.*, there "must be some articulated reasoning with some rational underpinning to support the legal conclusion of obviousness." 127 S. Ct. 1727, 1741 (2007). Neither Gupta nor Blank, either separately or in combination, teach or suggest all of the limitations of the recited claims.

Gupta relates to "a method, system, and product for coordinating parallel update for a global index of and indexed table." (Gupta, Abstract). "Techniques for maintaining a global index of a table during parallel data manipulations operations involve a coordinator process, data manipulation slaves and index update slaves. The coordinator process *distributes* data manipulation operations among a plurality of data manipulation slaves." (*Id.*, col. 8, ll. 1-6) (emphasis added). Gupta teaches a method using parallel DML ("PDML") operations that accomplishes the "need to update a global index as a result of PDML operations without suffering the deficiencies of lost clustering, or contention for the same block, the latter leading to excessive waits or block pinging." (*See id.*, col. 7, ll. 35-38).

Gupta teaches sorting maintenance records and determining a range by reading key values from the sorted maintenance records. (*See id.*, col. 15, ll. 35-67). A coordinator process then uses these ranges in distributing records to multiple slave processes. (*See id.*, col. 14, ll. 9-14). The slave processes use the maintenance records distributed by the coordinator process to update a global index. (*See id.*, col. 14, ll. 16-20).

Gupta fails to teach or suggest at least determining partition delimiters, each partition delimiter separating the table into non-overlapping partitions of records, each partition dedicated to one processing unit for index creation, wherein the step of determining comprises sampling the database table of records to determine an approximate distribution of at least one key value in the records; accessing the table records in parallel, wherein each processing unit accesses each of the records, wherein the step of accessing occurs after the step of determining; and filtering the accessed records in parallel, wherein each processing unit determines which records to keep, as recited in independent claim 1. As noted above, Gupta teaches updating a global index using multiple slave processes. In order to update the index, a coordinator process determines ranges for a set of maintenance records (a set of records that have been updated). The coordinator process then assigns the different ranges to various slave processes. After assigning the range, the coordinator process *distributes the maintenance records* to each slave process such that *each slave process receives only the maintenance records that fall within its assigned range.* (*See* Gupta, col. 14, ll. 9-20). Upon receiving the maintenance records, the slave processes use the maintenance records to update a global index. The teachings of Gupta require a great amount of work by the coordinator process to distribute maintenance records to each slave process.

Conversely, claim 1 describes determining partition delimiters, each partition delimiter separating the table into non-overlapping partitions of records, each partition dedicated to one processing unit for index creation, wherein the step of determining comprises sampling the database table of records to determine an approximate distribution of at least one key value in the records. After sampling the database to determine an approximate distribution used in determining partition delimiters, each processor undertakes the steps of accessing the table records in parallel, wherein each processing unit accesses each of the records and filtering the accessed records in parallel, wherein each processing unit determines which records to keep. The embodiment recited in claim 1 recites parallel index creation in which an entire table, not just maintenance records, can be indexed without having to undertake the effort of a coordinated distribution of records as taught in Gupta. The *single* coordinator process providing specific maintenance records to each of the multiple slaves is not the same as each processing unit accesses each of the records. There is no suggestion in Gupta that each of the slaves accesses each record in the table. Rather, each slave process in Gupta receives only the maintenance records within its range. This is fundamentally different from the approach of the method

described in claim 1 wherein each processing unit accesses each record, thereby eliminating the need for a coordinator process or other process to provide only the records within a partition to each processing unit.

Blank does not compensate for this deficiency. Blank relates to a "recover/build index system [that] builds an index for a file by scanning partitions of the file in parallel to retrieve key/rid values. The recover/build index system then sorts the scanned key/rid values for each partition in parallel." (col. 1, ll. 37-41). After the data is sorted in parallel, a "merge program merges the sort streams received from the sort programs to create a merge stream. The merge program accepts the sort stream from two or more sort programs. The merge program then passes the merge stream to an index build program." (col. 3, ll. 10-14). Thus, Blank teaches a method where a parallel sort is merged via combining the data streams produced by two or more sorts into a single data stream. Blank then performs *index creation on this single data stream.*

Blank does not teach determining partition delimiters, each partition delimiter separating the table into non-overlapping partitions of records, each partition dedicated to one processing unit for index creation, wherein the step of determining comprises sampling the database table of records to determine an approximate distribution of at least one key value in the records, as recited in claim 1. Indeed, Blank only indexes the single data stream because it fails to teach or suggest this step. Because no pre-sampling of data to determine non-overlapping partitions is taught in Blank, the individual sort streams *must* be merged *before* an index is created.

Furthermore, while multiple processing units are taught in Blank, the reference also teaches that each processing unit accesses only a portion of the table, i.e., each processing unit scans a single partition. Blank teaches,

> [t]he scan programs **108** executing in parallel extract key values (of a particular key) and record identifiers (rids) or pointers from the partitions **120** to create a key/rid or scan stream *for each partition* **112**. (Blank, col. 2, l. 64 – col. 3, l.1) (emphasis added).

The scan programs in Blank are only assigned a particular partition of the table, not each of the records in the table. Thus, Blank also fails to teach or suggest accessing the table records

in parallel and filtering the accessed records in parallel, wherein each processing unit determines which records to keep.

In any event, it would be improper to combine the parallel scan and sort taught by Blank with the system of Gupta. There "must be some articulated reasoning with some rational underpinning to support the legal conclusion of obviousness." *KSR Int'l Co. v. Teleflex, Inc.*, 127 S. Ct. 1727, 1741 (2007). Gupta is predicated on a central process or other process providing specific, partitioned records to slave processes. There is simply no need in Gupta for a parallel scan or sort by the slave process. In fact, Gupta specifically teaches away from using parallel processes to access *each record*:

> "Because each slave has a non-overlapping range, there is little or no contention for the same leaf block – thus latch contention and block pinging are drastically reduced." (Gupta, col. 14, ll. 20-23).

For at least these reasons, claim 1 is allowable over the cited references and Applicants respectfully request that the Examine issue a notice of allowance.

For at least similar reasons, both Gupta and Blank also fail to teach or suggest the other independent claims. For example, independent claim 14 recites, *inter alia*,

> a partition tool that determines partition delimiters, each partition delimiter separating the table into non-overlapping partitions of records, each partition dedicated to one processing unit for index creation, wherein the step of determining comprises sampling the database table of records to determine an approximate distribution of at least one key value in the records;
>
> a plurality of processing units that respectively accesses the database table in parallel, wherein each of the respective processing units accesses each of the records and filters the accessed records to determine which records to keep and wherein each of the respective processing units creates a sub-index of database table records resulting in a plurality of sub-indexes. . . .

Independent claim 18 recites, *inter alia*,

> determining partition delimiters, each partition delimiter separating the table into non-overlapping partitions of records, wherein the step of determining comprises sampling the database table of records to determine an approximate distribution of at least one key value in the records, and wherein at least one partition is dedicated to a first processing unit for index creation and at least one other partition is dedicated a second processing unit for index creation. . . .

Independent claim 20 recites, *inter alia*,

> determining partition delimiters, each partition delimiter separating the table into non-overlapping partitions of records, each partition dedicated to one processing unit for index creation, wherein the step of determining comprises sampling the database table of records to determine an approximate distribution of at least one key value in the records;
>
> accessing the table records in parallel, wherein each processing unit accesses each of the records, wherein the step of accessing occurs after the step of determining;
>
> filtering the accessed records in parallel, wherein each processing unit determines which records to keep. . . .

Finally, independent claim 24 recites, *inter alia*,

> a sampling module that samples the table of data records to determine sub-index delimiters, wherein the sub-index delimiters are used as partition delimiters separating the table into non-overlapping portions of records. . . .

For the forgoing reasons, neither Gupta nor Blank, alone or in combination, teach all of the limitations of independent claims 1, 14, 18, 20, and 24 and therefore cannot anticipate the present invention as claimed. Claims 1, 14, 18, 20, and 24 are allowable over the references of record and should be allowed. All other claims, *i.e.*, claims 2-13, 15-17, 19, 21-23, and 25-26 depend from one of the allowable independent claims and are, thus, also allowable over the prior art of record. Therefore, Applicants respectfully request that the Examiner issue a notice of allowance, for all claims, at his earliest convenience.

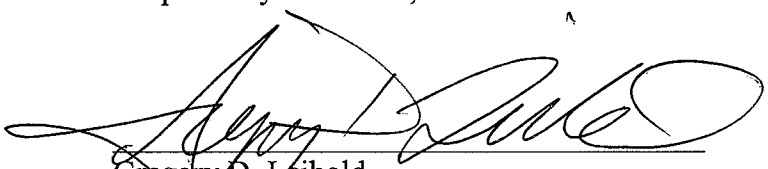
Conclusion

This Amendment fully responds to the Office Action mailed on June 28, 2007. Still, that Office Action may contain arguments and rejections that are not directly addressed by this Amendment due to the fact that they are rendered moot in light of the preceding arguments in favor of patentability. Hence, failure of this Amendment to directly address an argument raised in the Office Action should not be taken as an indication that the Applicants believe the argument has merit. Furthermore, the claims of the present application may include other elements, not discussed in this Amendment, which are not shown, taught, or otherwise suggested by the art of record. Accordingly, the preceding arguments in favor of patentability are advanced without prejudice to other bases of patentability.

It is believed that no further fees are due with this Response. However, the Commissioner is hereby authorized to charge any deficiencies or credit any overpayment with respect to this patent application to deposit account number 13-2725.

In light of the above remarks and amendments, it is believed that the application is now in condition for allowance, and such action is respectfully requested. Should any additional issues need to be resolved, the Examiner is respectfully requested to telephone the undersigned to attempt to resolve those issues.

Respectfully submitted,



Date: January 15, 2008

Gregory D. Leibold
Reg. No. 36,408
MERCHANT & GOULD P.C.
P.O. Box 2903
Minneapolis, Minnesota 55402-0903
(303) 357-1642